
# South African Computer Olympiad
# Training camp 3, 2004
# Day 2

## Overview

| **Author** | **Marco Gallotta** | **Nick Pilkington** | **Richard Starfield** | **Linsen Loots** |
|---|---|---|---|---|
| **Problem** | **queue** | **tiles** | **numbers** | **notes** |
| Source | queue.pas<br>queue.c<br>queue.cpp | tiles.pas<br>tiles.c<br>tiles.cpp | numbers.pas<br>numbers.c<br>numbers.cpp | notes.pas<br>notes.c<br>notes.cpp |
| Input file | queue.in | tiles.in | numbers.in | notes.in |
| Output file | queue.out | tiles.out | numbers.out | notes.out |
| Time limit | 1 second | 1 second | 1 second | 1 second |
| Number of tests | 10 | 10 | 10 | 10 |
| Points per test | 10 | 10 | 10 | 10 |
| **Total points** | **100** | **100** | **100** | **100** |

The maximum total score is 400 points.


# South African Computer Olympiad
# Training camp 3, 2004
# Day 2

## Queue

### Author

Marco Gallotta

### Introduction

The Guji tribe have recently discovered cinemas. This is the first time the Guji tribe has ever had to form queues. The Guji chief noticed one day while inspecting these new queues that all four possibilities of 2 successive males and females (mm, mf, fm, ff) were present. Of course, some of the combinations overlapped others.

### Task

Given $N$, find the minimum length queue of people such that every combination of $N$ successive male and female occurs in that queue.

### Example

The case $N = 2$ can be satisfied by a queue of length 5 as follows:

male, female, female, male, male

As you can see, all possible pairs of males and females appear in the queue.

### Input (queue.in)

The only line of input will contain a single integer $N$.

#### Sample input

```
2
```

### Output (queue.out)

The first line of output must contain a single integer $S$, the length of the shortest queue that you can find. The next line must contain a combination of the letters 'm' and 'f'. This line represents the people in the queue, with an 'm' representing a male in the queue and an 'f' representing a female in the queue.

#### Sample output

```
5
mffmm
```

### Constraints

- $2 \leq N \leq 17$

#### 50% constraints

- $2 \leq N \leq 6$

### Time limit

1 second.

### Scoring

- If the formatting of your solution is in any way different from the way described above, you will receive 0.
- If the length of your queue is greater than 200000, you will receive 0.
- If you have left out any of the combinations in your queue, you will receive 0.
- If the length of the queue equals or betters the optimal solution, you will receive 10.
- Otherwise, if the length of your queue is $S$ and the optimal length is $O$, you will receive $\frac{4O}{S}$ rounded down to the nearest integer.


# South African Computer Olympiad
# Training camp 3, 2004
# Day 2

# Sliding Tiles, Again!

## Author

Nick Pilkington

## Introduction

Not too long ago an unnamed IOI squad was given a practice task of solving the famous sliding tile puzzle. Unfortunately the team didn't perform all that well in the task and wild rumors of perversely obscure test data ran riot after the competition. Of course not doubting their own abilities as programmers the squad suspected foul play on the part of the problem author. They have managed to get hold of the data used and now wish to ascertain whether or not it is possible to solve the problem at all!

A sliding tile puzzle is simply a square grid of tiles, with one tile missing. The tiles have the numbers 1 to $N^2 - 1$, and the goal of the puzzle is to use the blank space to move the tiles around into a specific order. In particular, the tiles must be in ascending order starting from the top-left corner, moving from left to right and then continuing at the left side of the next row down. Finally the empty tile should be in the bottom right corner. See Table 1 for a graphical illustration:

| | | |
|---|---|---|
| 1 | 2 | 3 |
| | 4 | 6 |
| 7 | 5 | 8 |

Table 1: A graphical illustration of the sliding tile puzzle. This is solved by first moving tile 4, then tile 5, and finally tile 8.

## Task

You will be given the initial state of the sliding tiles puzzle and will have to determine whether or not the puzzle can actually be solved by some finite number of legal moves.

## Example

Consider the sliding tile puzzles in Table 2.

The left puzzle can be solved; however, the right one cannot be solved!

| | | | |
|---|---|---|---|
| 1 | 2 | 1 | 3 |
| | 3 | 2 | |

Table 2: An example

## Input (tiles.in)

Line 1: Two integers $T$, $N$. $T$ represents the number of squares to follow in the file (note there is possibly more than one square to solve in each file). N represents the dimensions of all of the squares to follow.

Line 2..$N \times T + 1$: Each line contains $N$ integers representing a row of tiles (0 indicates the blank)

### Sample input

```
1 3
1 2 3
4 5 6
7 0 8
```

## Output (tiles.out)

The output file should contain a single line containing a bit string of length $T$. Each character should refer to the square of the same position. A 1 should indicate that the case is possible, and a 0 that it is not.

### Sample output

```
1
```

## Constraints

- $1 \leq T \leq 5$
- $1 \leq N \leq 150$

### 50% constraints

- $1 \leq T \leq 3$
- $1 \leq N \leq 4$

## Time limit

1 second.


# South African Computer Olympiad
# Training camp 3, 2004
# Day 2

## Scoring

If the output is in any way invalid you will score 0%. If it represents the correct bit string for the input data you will score 100% and will have redeemed the pride of the unnamed squad!

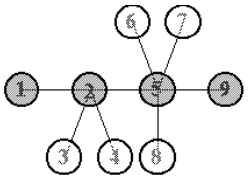

Figure 1: Example of hut layout

# Hut Numbering

## Author

Richard Starfield

## Introduction

A small group of islanders have decided that, as part of the modernisation of their village, every hut must be labelled with a unique number. Naturally, the village also has a number of paths, each connecting two huts, which the villagers would like to label distinctly. To simplify matters though, they want each path's number to be the absolute difference between the numbers of the two huts it connects.

The huts are connected to form a special spanning tree. A certain group of huts forms a central chain, where each hut is connected to the next. Any hut not on this chain is connected directly to another hut on the chain by a single path.

## Task

Given the number of huts $N$ and the set of $P$ paths between them, you must label every hut with a unique number such that the absolute difference between any two directly connected huts is unique. This will ensure that every path has a unique, easily calculated number.

## Example

## Input (numbers.in)

The first line contains a single integer, $N$, the number of huts in the village. Lines 2–$N$ each contain two space separated integers, $H_1$ and $H_2$, specifying a path connecting these two huts.

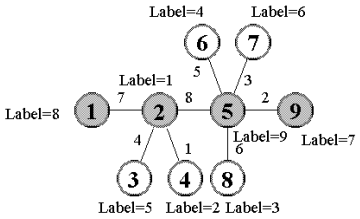


Figure 2: Example of labelling

### Sample input

```
9
1 2
6 5
5 7
4 2
2 3
8 5
2 5
5 9
```

## Output (numbers.out)

Lines 1 . . . $N$ each contain a single integer $H_i$, the label associated with hut $i$. Extraneous whitespace will no affect the validity of a solution.

### Sample output

```
9
1
8
7
6
2
3
4
5
```

## Constraints

- $2 \leq N \leq 10000$

### 50% constraints

- $2 \leq N \leq 100$

## Time limit

1 second.


# South African Computer Olympiad
# Training camp 3, 2004
# Day 2

## Scoring

- Incorrectly formatted outputs will score 0.
- Solutions which do not assign a unique number to every hut will score 0.
- Solutions where the absolute difference between any pair of directly connected huts is not uniquem will score 0.
- Only correct solutions will score 10 points.

# Bank Notes

## Author

Linsen Loots

## Introduction

Having lived a long and successful life, John has decided to give his possessions to his many sons. His most prized possession is his collection of Binarian Dollar notes. These are notes in the currency of Binaria, which only has denominations of powers of 2, and are very valuable.

## Task

Because his sons are highly competitive, John needs to distribute the notes in an absolutely fair manner. He has asked you to find out how much money will be left if he gives away as much as possible so that each son will receive exactly the same amount. The notes cannot be broken into smaller denominations.

## Example

John has 3 sons, and 6 notes to share out: one \$1-note, four \$2-notes and one \$4-note. The best way of distributing these are to give each son \$4 (\$4 to one, two \$2 to each of the others), leaving \$1 that is not given away.

## Input (notes.in)

The first line of the input file will contain a single integer $N$, the number of sons John has. The next line will contain a single integer $M$, the number of notes in John's collection. The third line will contain $M$ space-separated integer, the values of the respective notes.

### Sample input

```
3
6
2 1 2 4 2 2
```

## Output (notes.out)

The output should be a single integer $L$, the total amount of money left over after as much as possible has been given to the sons, with each son receiving an equal amount of money.

### Sample output

```
1
```

## Constraints

- $1 \leq N \leq 20$
- $1 \leq M \leq 64$
- $1 \leq V_i \leq 2^{24}$, where $V_i$ is the value of note i

### 50% constraints

- $1 \leq N < 10$
- $1 \leq M \leq 50$
- $1 \leq V_i < 2^{15}$

## Time limit

1 second.

## Scoring

10 marks for a correct answer, otherwise zero.